14
7-30-01

Docket No.: 42390.P2319R

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the application of:	)	
	)	
Marisetty	)	
	)	
Serial No.: 09/224,620	)	Examiner: Auve, G.
	)	
Reissue of US Patent 5,590,342	)	
	)	
Filed: 12/31/98	)	Art Unit: 2781
	)	
For: Method and Apparatus for Reducing Power Consumption in a Computer System Using Virtual Device Drivers	)	

Hon. Commissioner of
Patent & Trademarks
Washington, D.C. 20231

DECLARATION UNDER 37 CFR § 1.131

I, Suresh K. Marisetty, declare that I am an inventor of the invention ("present invention") defined in all claims of the above-identified patent application ("present application"), and all of the events described or mentioned in this declaration occurred in the United States.

I further declare the following:

The present invention was conceived prior to April 1, 1994 (the "effective date"). This is evidenced by the written description of the present invention prepared and dated in an invention disclosure form (Exhibit 1) evidencing possession of the invention prior to the effective date.

Soon thereafter, the invention disclosure form was diligently submitted to an Intel (the Assignee) intellectual property committee for review. Soon thereafter, the invention disclosure form was reviewed by the committee, and the committee decided to file a patent application covering the present invention. A patent attorney was diligently contacted by the committee and was tasked with preparing the present application.

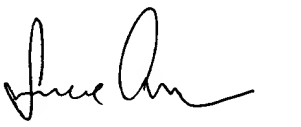
During this time, I was also diligently working to reduce the present invention to practice, by building and constructing a test system (See Exhibit 2, a paper, which under "challenges" indicates that implementation "is in progress". Although this paper is undated, it was faxed to applicant's attorney on 8-30-94 and cites a reference dated April 1994. Thus, the paper in Exhibit 2 corroborates that actual reduction to practice was in progress at some point between April and August 1994).

An initial draft of the present application was prepared by the patent attorney and forwarded to me for my review. After at least one iteration of diligent review by me and revision by the patent attorney, incorporating my comments, a final draft of the present application was provided to me for my signature. Soon thereafter, the present application was filed.

Thus, at all times between the effective date and the filing date of the Original Application on November 29, 1994, either I was diligently working to reduce the invention to practice, or I was diligently working with a patent attorney to file the patent application, or both.

I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with knowledge that willful false statements and the like so made are punishable by fine or imprisonment or both under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Respectfully submitted,



Suresh K. Marisetty

Exhibit 1: Invention Disclosure Form

FIRST CLASS CERTIFICATE OF MAILING
(37 C.F.R. § 1.8(a))

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Commissioner of Patents and Trademarks, Washington, D.C. 20231

on July 17, 2001.
Date of Deposit

Kelli Frey
Name of Person Mailing Correspondence

Kelli Frey — 7-17-01
Signature — Date

INTEL INVENTION DISCLOSURE

INTEL CONFIDENTIAL MCG #2964 DATE: 2/28/94

It is important to provide accurate and detailed information on this form. The information you provide will be used to evaluate your invention for possible filing as a patent application. When compllleted, please return this form to Legal. Do not leave any blanks. If you have any questions regarding this form, please call 765-1986 or 696-2033

1. Inventor(s):
   a. Name: Suresh Marisetty
   b. Empl No: 71756 Grp/Div/Dept: 7336-6 Ext: 55181 MS: SC9-25
   c. Home Address: 1662 Lederer Circle, San Jose, CA 95131
   d. Citizenship: India Supervisor: Graham Kirby Ext: 5-4505 MS: SC9-25
   e. Check One: MPG ___ MCG ___ x ISG ___ Other ___

   a. Name:
   b. Empl No: ___ Grp/Div/Dept: ___ Ext: ___ MS: ___
   c. Home Address:
   d. Citizenship: US Supervisor: ___ Ext: ___ MS: ___
   e. Check One: MPG ___ MCG ___ ISG ___ Other ___

   a. Name:
   b. Empl No: ___ Grp/Div/Dept: ___ Ext: ___ MS: ___
   c. Home Address:
   d. Citizenship: ___ Supervisor: ___ Ext: ___ MS: ___
   e. Check One: MPG ___ EPG ___ x ISG ___ Other ___

2. Title of Invention: **Power Management in Notebooks and Green PC's using Windows VxD's**
   a. Has a description of your invention been, or will shortly be, published?
      NO ___ YES x Date ___
   b. Has your invention been used/sold or planned to be used/sold by Intel or others?
      NO ___ YES x Date Soon (tbd)

3. Please use the remaining space and the back of this page to provide an abstract of your invention. This information will be used to evaluate your invention for possible filing as a patent application. Therefore, please include the following information in your abstract:

   a. State the general purpose(s) of your invention;
   b. Describe advantage(s) of your invention over what is done now;
   c. Describe essential element(s) that are key to your invention; and
   d. Value of your invention to Intel (how will it be used?).

LEGAL OK
RCC 3/23/94

***HAVE SUPERVISOR READ AND SIGN COMPLETED FORM.**

SF0004/7-9-91 **SUPERVISOR:** _____

Power Management in Notebook and Green PC's With Windows Virtual Device Drivers

Patent Disclosure

Author:
Suresh Marisetty
MCG Software Architecture
Intel Corporation
msuresh@mipos3.intel.com

Date: 2/28/93
2200 Mission College Blvd.
Santa Clara, CA 95052
MS: SC9-25
(408) 441-0712

ABSTRACT:

The advent of Microsoft Windows for PC-AT machines has and is creating exciting and interesting opportunities for software developers. The introduction of hardware abstraction layers in Windows is opening the doors for myriad of possibilities. An example of this in Windows are GDI and Virtual Device Drivers (VxD's).

The Windows VxD's have capabilities that are so powerful, they could literally solve almost any problem that hardware designers, driver developers, application developers and the OS developers are faced with. ***VxD's are playing handyman in the exciting world of Windows.***

This present paper describes one such opportunity, where VxD can play a very important role in Power Managing an Intel Architecture PC System. This paper also identifies how rapid changes in the software technology is making the hardware designers to rethink about their traditional design concepts and methodologies.

This paper describes a technique which provides an enhanced & fine grain power management scheme in Notebook and Green PC's using Microsoft Windows Virtual Device Drivers(VxD's) and its powerful IO/Interrupt/VxD Trapping capabilities, in a OS friendly manner. This technique also describes how it reduces the dependency and need for system specific power-management HARDWARE and makes it manageable by SOFTWARE. In addition to the many benefits described above, the use of VxD also reduces the latencies in a system by using the Protected Mode operation of an Intel Architecture CPU. This new technique also outlines how it overcomes some of the short comings of the traditional hardware & software power-management techniques.

This new mechanism shows how to power-manage a system WITH TRADITIONAL applications and device drivers, which are not Power-Aware and illustrates a clear distinction between the existing hardware and software methodologies like Advance Power Management Spec. (APM 1.1). It also describes how VxD based software power-management technique will allow adaptability to a dynamic environment like Plug-n-Play with ease.

Towards the end, this paper will summarize the Pros & Cons of this technique and the impending threats that the rapidly evolving software technology could pose to the Intel Based Architecture's in general and the SMM architecture in particular.

Purpose of Invention:

The purpose of this invention is to provide an enhanced power management scheme in Notebook and Green PC's Using Microsoft Windows Virtual Device Drivers(VxD's). The new power management technique reduces the dependency and need for system specific power-management HARDWARE and makes it manageable by SOFTWARE. The new technique also eliminates some the short comings of the traditional hardware power-management techniques.

The new mechanism allows to power-manage a system WITH TRADITIONAL applications and device drivers. The applications and device drivers need not be POWER-AWARE. The software power-management technique also allows adaptability to a dynamic environment like Plug-n-Play with great ease.

Background Information:

The existing power-management techniques in a typical notebook PC use system specific hardware mechanisms to provide fine grain power management. These hardware mechanisms use processor specific Interrupts (System Management Interrupt) and other System Activity Monitoring Hardware to provide a reasonable amount of power conservation. In addition to the hardware mechanism, some software mechanisms are used to detect the CPU idle conditions to put the system in power-conservation modes (Advance Power Management).

Existing Hardware Mechanism:

In the existing architecture's, there are four different types of power management modes and are as follows:

- Fully On
- Local Standby
- Global Standby or Fully Off (Vcc Suspend)
- Suspend or Hibernation (0 v Suspend)

Power Management is used to reduce the Dynamic and Static Power consumption of a system to increase the battery life (Mobile PC) and to reduce the energy cost (Green PC). Dynamic Power is consumed by all components during state switching of internal electronic circuits, while Static Power is consumed due to leakage currents of an electronic devices. The different components of a system can be power-managed are as follows:

- CPU
- Generic IO Controllers
- Graphics System
- Hard Disk System
- Floppy Disk System
- Serial Interfaces & associated devices (MODEM, MOUSE, Printers, LAN)

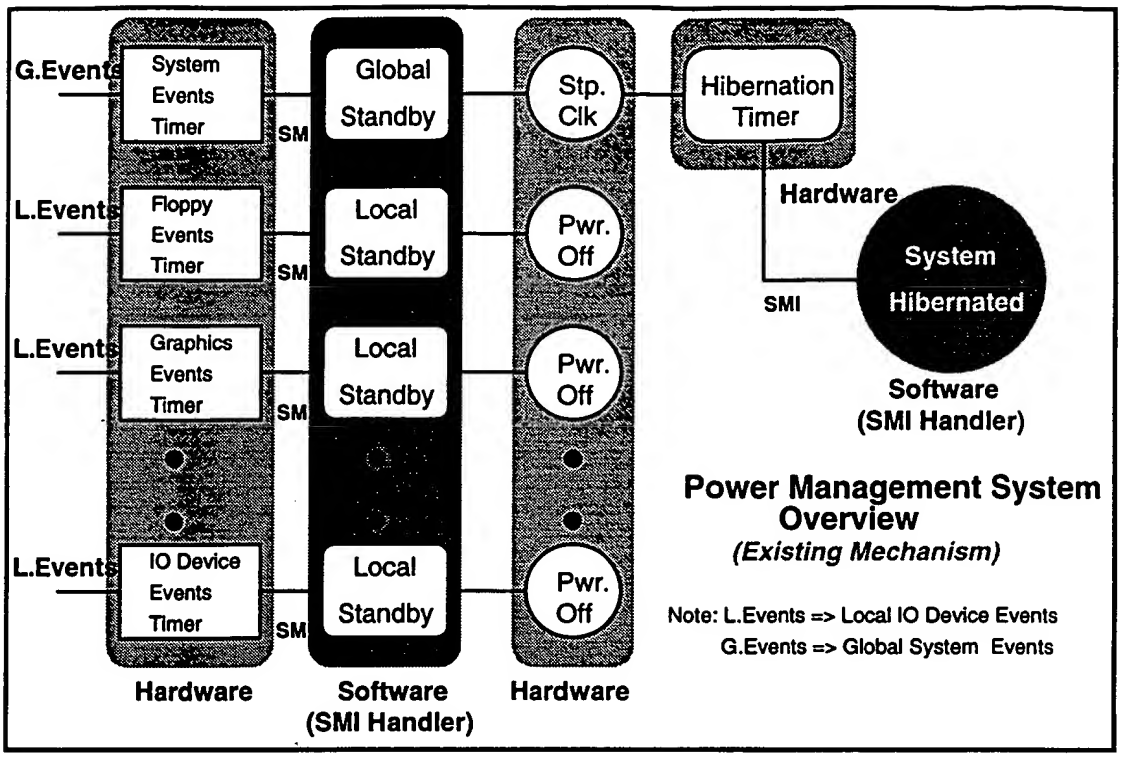
- Parallel Interfaces (Printers, External SCSI and etc)
- Keyboard Devices
- LAN Interfaces
- DRAM Memory System

Fully On: In this condition, all the components of a typical system are fully powered. In this condition all the clocks in the system will be running at either full speed or slow speed.

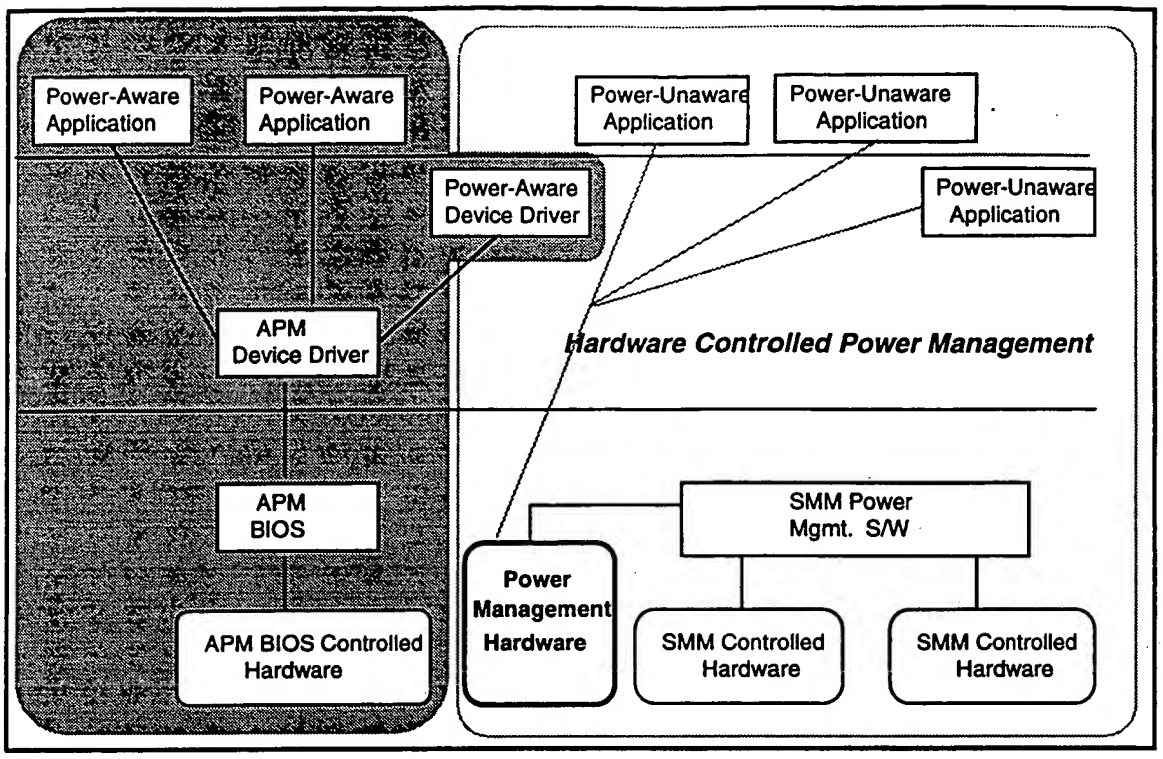
Local Standby: In this condition, certain temporarily unused LOCAL devices in the system, like a Floppy Device, Graphics Device (LCD, CRT), Harddisk Device and Others are powered down. The power to these turned-off devices is restored when the internal or external system events require the services of these devices. The system maintains idle timers for each of these power-manageable devices. The idle timers enter an expired TIME-OUT state when it detects idleness on this interface after a pre-defined period of time.

Global Standby: In this condition, most of the devices in the system are powered down, except the CPU and the system memory DRAM. The clock to the CPU is Stopped with the DRAM memory operating in an extended power conservation mode (also called Standby with self refresh). The CPU and DRAM at this point in time is in Standby Mode and is ready to be activated when a system event occurs. These events can be a keyboard/mouse clock or other System Interrupts (IRQ0-IRQ15, NMI, SMI and etc).

Hibernation: Hibernation is a mode, where the system is put in a shut-down condition. Here the complete systems state is saved to the harddisk due to User or System Initiated Hibernation Mode. When a system detects an idle condition after a predetermined period of time in Global Standby, it can initiate a transfer to the Hibernation mode. When the system is turned back into action, the Hibernation mode will restore the system back to exactly the same state as it was before, when it went into Hibernation.



Event Detection: As can be seen from the illustration, the Power Management System Requires the detection of Local Events and Global Events. In the existing mechanism, this job is handled by special hardware or Power Aware Applications and Device Drivers. This hardware traps or snoops on IO cycles to the IO Devices. These IO cycles are directed to a Specific Set of Unique IO Addresses (with CPU IN and OUT instructions) for each IO Device. These IO addresses are always Static and are know at the system boot-up time. These IO addresses do not change over the life time of the current System Login. In certain implementations these IO addresses are programmable in the IO Hardware and in others are fixed in Hardware. The deterministic nature of the mapping of the IO Space to the IO devices (as per IBM PC-AT/DOS standards), makes it easy to design Standard hardware which is consistent across all PC DOS platforms.



Problem1: One of the disadvantage with the existing mechanism is the use of Hardware. As can be seen from the above illustration, each of the IO Device needs a an Idle Timer to monitor the activity. ***This imposes a great deal of restriction on how many Hardware Timers can be designed into the system.*** Also most implementations Hardcode the IO Trapping address of the IO devices to save "Gates". ***This makes the system more expensive and in fact the additional hardware added will consume more power to support itself.***

Problem2: The Second big the disadvantage with the existing mechanism is that it assumes that all IO devices use Standard IO Address over the life time of the system, i.e. Static IO Address Mapping. ***This in fact places a Severe Restrictions on the usage of the System Resources and Demands a Perfect Hardware Compatibility.***

Problem3: The Third problem is that the software which does Power Management in the traditional systems is completely de coupled from the Operating System and Applications. This makes the systems prone to Blind Spots, i.e. OS doing things and Pwr. Mgmt Software doing things, with neither of them aware of the activities going in the other worlds. This will lead to problems like system crashes (ex: when Pwr. Mgmt. Software interrupts and takes away the control from OS, when it is executing a critical section of its code).

Important Advantage That should be applicable to other types of drivers

PNP Devices use this solution

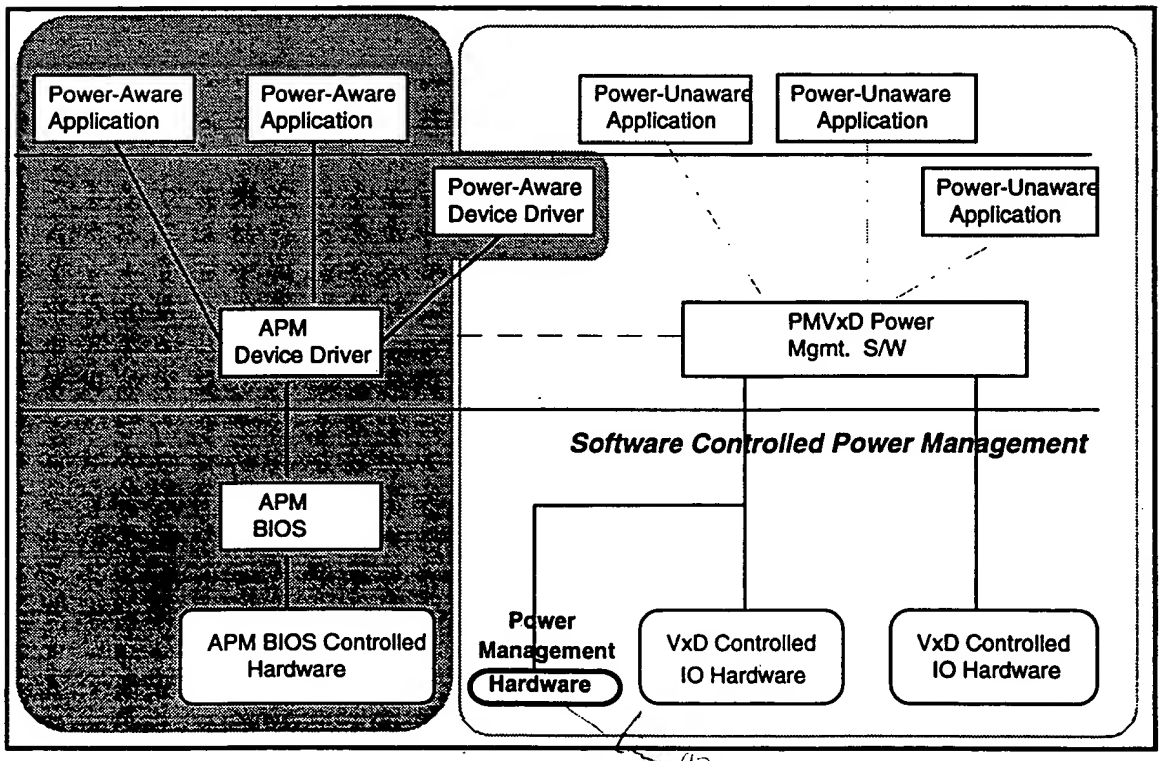
Problem4: The Fourth problem is that the current generation of Device Drivers & Operating Systems Virtualize IO Ports. When the IO ports are virtualised, it becomes difficult and in some cases impossible for the Pwr. Mgmt. software and hardware to detect it. This will lead to the Pwr. Mgmt system to monitor and trap on invalid IO device Addresses and generate improper events in the system.

Problem5: The Fifth problem is that even if the above problems are circumvented, the

assumption of IO Device Addresses are no longer deterministic or visible to the Pwr. Mgmt. Software at system boot-up time. Since the current and future generations of Operating Systems will resort to Plug-n-Play Architecture, where the IO device address can and will change Dynamically during the life time of the current system boot. If and when these Dynamic Remapping of IO devices occurs, there is no easy way to communicate it and remap the IO Device Addresses in the Pwr. Mgmt. Hardware and Software.

Problem6: There are certain software power-management techniques that are in existence (APM 1.1 Spec.). The APM spec. assumes that the Applications and the associated Device Drivers in the system be APM aware to monitor and/or control power management. This makes it difficult to manage a system with applications and drivers which are not Power Aware.

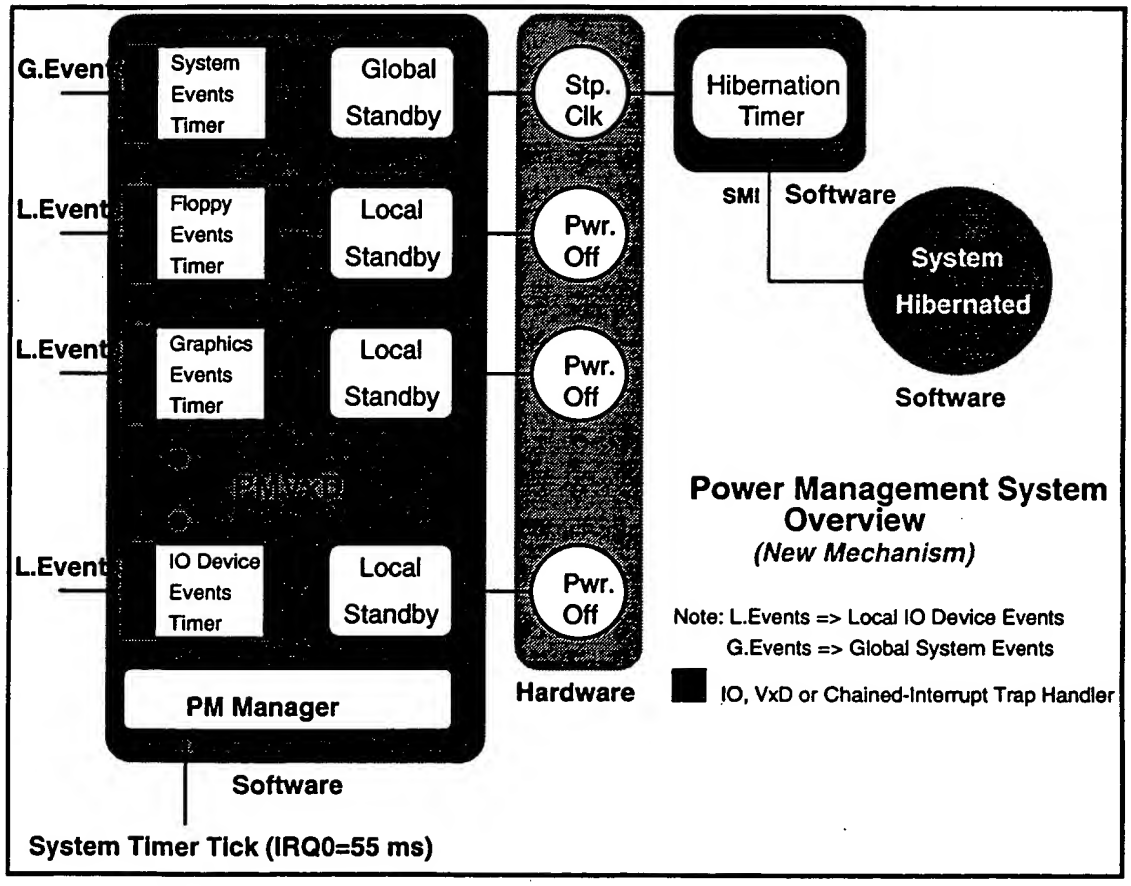
Solution: The solution to most of the above shortcomings of the Power Management Techniques is to provide most of the support discussed above in Software by Windows Virtual Device Drivers (VxD's) and its IO and Interrupt Trapping Capabilities. The VxD work in close co-operation with the Operating System and at all time are aware of the dynamic changes in the system and can adapt to it in a seamless fashion. In fact, the above VxD can make all Windows Applications and Device "Virtual Power Aware" and communicate with the APM 1.1 interface on the their behalf.



Description: The new technique consist of a Power Management Virtual Device Driver (PMVxD) and a bunch of Data Structures. The Data Structures are initialized at system boot-up time to provide Command/Status information to the PMVxD. The PMVxD consists of several idle timers, one for each enabled IO Device solution to most of the

above shortcomings of the Power Management Technique is to provide most of the support discussed above in Software by Windows Virtual Device Drivers.

The PMVxD is chained into the system timer Interrupt (IRQ0), which ticks every 55 ms. This is used as a time-base for the PMVxD. The PMVxD uses the IRQ0 tick to monitor the activity of each IO Device and also Overall System Activity. The PMVxD Installs handlers for IO Trapping for each IO Device IO address Range. Anytime the IO address is trapped, the counter is updated (increment/decrement) to reflect the activity status. For IO devices, whose IO Address is Virtualized, Interrupt Trap handlers are chained into the Interrupt Handlers for that specific IO device. This Interrupt Trap Handler maintains the idle timers for the IO device. An example of this is the Software Interrupt Trap Handler for 0Eh, which is used for Floppy Disk IO accesses.

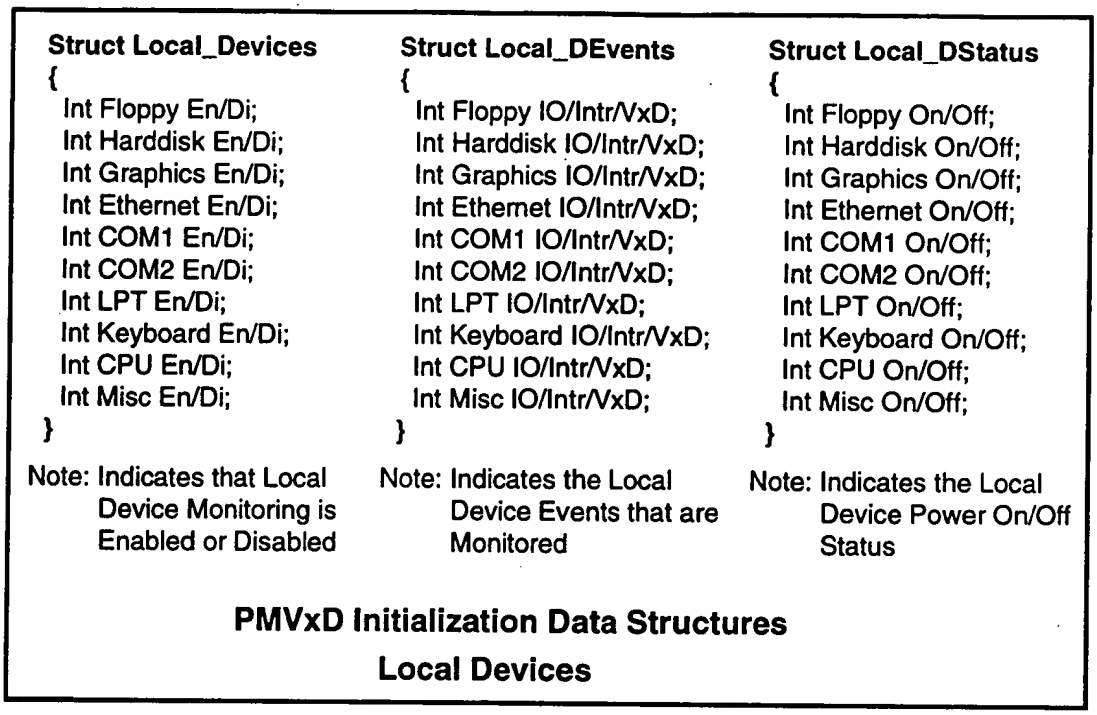


As illustrated above, the IO Events Timer can be a simple IO Trap Handler, a Device Driver Hook Handler or a Chained Interrupt Trap Handler incrementing/decrementing a software timer. The PM Manager (VxD) inspects the IO Event Timers during each IRQ0 tick from the system timer, in fact the inspection time can be an integral multiple of IRQ0 tick and can be specified by the PMVxD initialization data structures. When an IO Event Timer times-out, the PMVxD can turn off the power to the IO Device, through some system or device specific hardware mechanisms.

The PMVxD is part of the OS and appears as a device driver in the system. In a Plug-n-Play environment, when the System Resources are remapped to the IO Devices, the PMVxD is informed of the changes by the OS specific Configuration Manager or

Resource Manager. This clean communication mechanism between the OS and PMVxD will allow it to dynamically adapt itself to the changes gracefully.

The following figure illustrates typical data structures of a PMVxD:



Struct Local_Devices
{
Int Floppy En/Di;
Int Harddisk En/Di;
Int Graphics En/Di;
Int Ethernet En/Di;
Int COM1 En/Di;
Int COM2 En/Di;
Int LPT En/Di;
Int Keyboard En/Di;
Int CPU En/Di;
Int Misc En/Di;
}

Note: Indicates that Local Device Monitoring is Enabled or Disabled

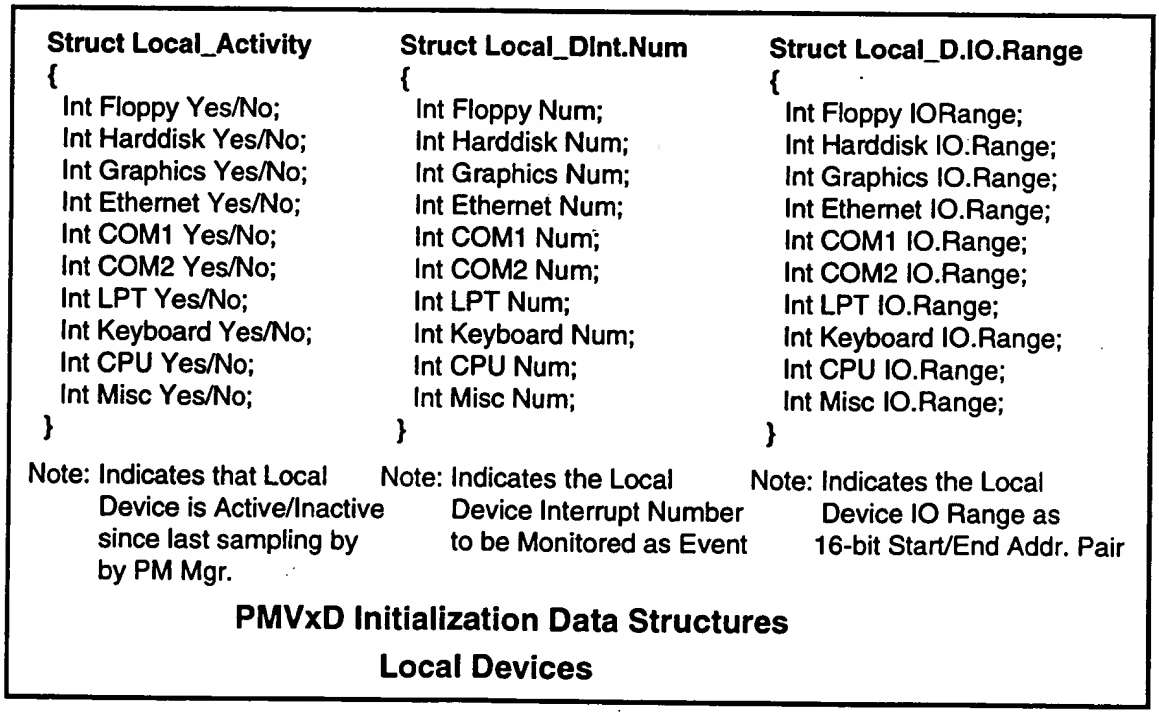
Struct Local_DEvents
{
Int Floppy IO/Intr/VxD;
Int Harddisk IO/Intr/VxD;
Int Graphics IO/Intr/VxD;
Int Ethernet IO/Intr/VxD;
Int COM1 IO/Intr/VxD;
Int COM2 IO/Intr/VxD;
Int LPT IO/Intr/VxD;
Int Keyboard IO/Intr/VxD;
Int CPU IO/Intr/VxD;
Int Misc IO/Intr/VxD;
}

Note: Indicates the Local Device Events that are Monitored

Struct Local_DStatus
{
Int Floppy On/Off;
Int Harddisk On/Off;
Int Graphics On/Off;
Int Ethernet On/Off;
Int COM1 On/Off;
Int COM2 On/Off;
Int LPT On/Off;
Int Keyboard On/Off;
Int CPU On/Off;
Int Misc On/Off;
}

Note: Indicates the Local Device Power On/Off Status

PMVxD Initialization Data Structures
Local Devices



Struct Local_Activity
{
Int Floppy Yes/No;
Int Harddisk Yes/No;
Int Graphics Yes/No;
Int Ethernet Yes/No;
Int COM1 Yes/No;
Int COM2 Yes/No;
Int LPT Yes/No;
Int Keyboard Yes/No;
Int CPU Yes/No;
Int Misc Yes/No;
}

Note: Indicates that Local Device is Active/Inactive since last sampling by by PM Mgr.

Struct Local_DInt.Num
{
Int Floppy Num;
Int Harddisk Num;
Int Graphics Num;
Int Ethernet Num;
Int COM1 Num;
Int COM2 Num;
Int LPT Num;
Int Keyboard Num;
Int CPU Num;
Int Misc Num;
}

Note: Indicates the Local Device Interrupt Number to be Monitored as Event

Struct Local_D.IO.Range
{
Int Floppy IORange;
Int Harddisk IO.Range;
Int Graphics IO.Range;
Int Ethernet IO.Range;
Int COM1 IO.Range;
Int COM2 IO.Range;
Int LPT IO.Range;
Int Keyboard IO.Range;
Int CPU IO.Range;
Int Misc IO.Range;
}

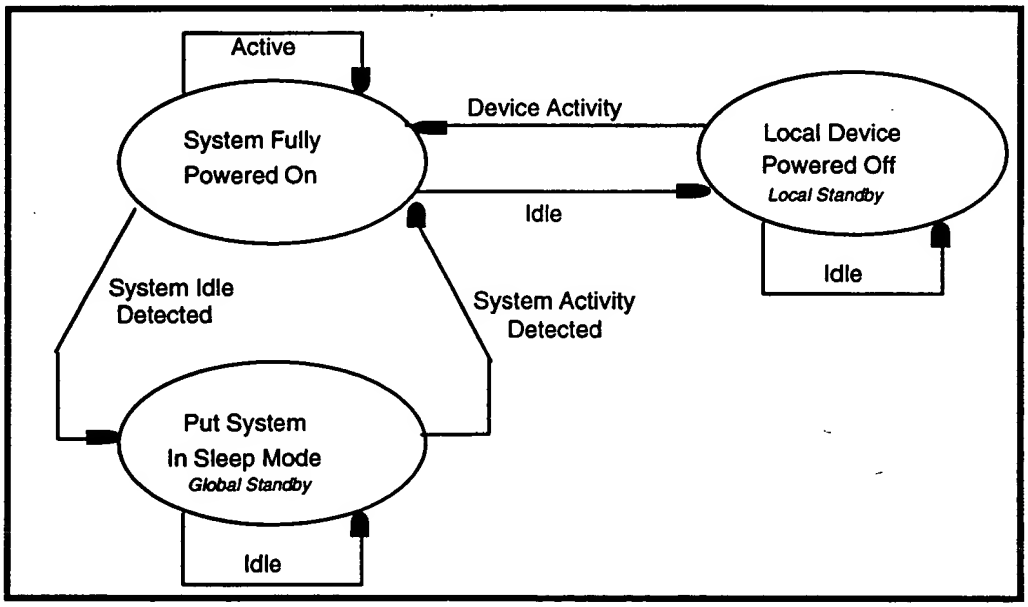
Note: Indicates the Local Device IO Range as 16-bit Start/End Addr. Pair

PMVxD Initialization Data Structures
Local Devices

Struct Global_Sys.Events	Struct Sys_Break.Events	Struct Suspend_Status
{	{	{
Int APM_Msg. En/Di;	Int APM_Msg. En/Di;	Int Local_Standby On/Off;
Int NMI En/Di;	Int NMI En/Di;	Int Global_Standby On/Off;
Int RING En/Di;	Int RING En/Di;	Int Fully_On On/Off;
Int IRQ<0:15> En/Di;	Int IRQ<0:15> En/Di;	Int Hibernation On/Off;
Int Misc En/Di;	Int Misc En/Di;	Int Misc On/Off;
}	}	}
Note: Indicates that System Events Monitoring is Enabled or Disabled	Note: Indicates that System Events Enabled as Break Events	Note: Indicates the System Power Miser Mode Status

PMVxD Initialization Data Structures
Global Events

The following figure illustrates the Power Management States the PMVxD will come accross in a system.



Claims:

- Makes use of the PMVxD's (Power Management Virtual Device Drivers) IO Trapping Capabilities to trap on IO Device Accesses for System/Device Idle Detection.
- Makes use of the PMVxD's Interrupt Trapping Capabilities to trap on IO Device Accesses for System/Device Idle Detection, when the IO ports are Virtualized.
- Uses a combination of IO Trapping and Interrupt Trapping mechanisms in the PMVxD's to Monitor the System and IO Device Activities.
- The PMVxD's works in close co-operation with the Operating System is always aware of the dynamic changes in the IO device state changes.
- The current mechanism uses the Standard PC Interrupt Mechanism for Pwr. Mgmt. and is not dependent on a CPU/System specific Power Management Interrupts.
- Since the PMVxD's is a software entity, it allows more room for expansion without

any hardware penalty. It is PMVxD is easily maintainable and scalable.

- Since most of the Power Mgmt. Hardware is eliminated, the new technique is power a saver.
- The PMVxD eliminates the system crash problems by eliminating the blind spots.
- The PMVxD power manages a system which has Traditional Apps and Device Drivers (one which are not power-aware).
- The PMVxD makes the Non-Power Aware Windows Applications and Device Drivers "Virtually Power Aware".

Docket No.: 42390.P2319R

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the application of:	
Marisetty	
Serial No.: 09/224,620	Examiner: Auve, G.
Reissue of US Patent 5,590,342	
Filed: 12/31/98	Art Unit: 2781
For: Method and Apparatus for Reducing Power Consumption in a Computer System Using Virtual Device Drivers	

Hon. Commissioner of
Patent & Trademarks
Washington, D.C. 20231

REGARDING DECLARATION UNDER 37 CFR § 1.131 & POWER MANGEMENT CONTROLLER API SPECIFICATION

Dear Sir:

Applicant submits herewith US Patent 5,620,022 and a "Power Management Controller API Specification", dated April 1994 listed on an Information Disclosure Statement. Applicant does not admit that the Power Management Controller API Specification was public prior in April of 1994, and reserves the right to contest whether this specification was indeed public at that time.

Nonetheless, applicant has submitted a declaration to swear behind both US Patent 5,620,022 and the Power Management Controller API Specification.

Applicant does not concede that it is necessary to do so, but rather provides this declaration in order to expedite prosecution in the present reissue application.

Respectfully submitted,

Date: 7/16/01

Jeffrey S. Draeger, Reg. No. 41,000
Direct Phone: 408-765-5935

FIRST CLASS CERTIFICATE OF MAILING
(37 C.F.R. § 1.8(a))

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage in an envelope addressed to the Commissioner of Patents and Trademarks, Washington, D.C. 20231

on July 17, 2001.
Date of Deposit

Kelli Ivey
Name of Person Mailing Correspondence

Kelli Ivey — 7-17-01
Signature — Date